Interleaving semantics and action refinement in event structures

Weidong Tang[1, 2, 3], Jinzhao Wu[2, 3*], Meiling Liu[2, 4]

[1]*Chengdu Institute of Computer Applications, Chinese Academy of Sciences, Chengdu 610041, China*

[2]*School of Information Science and Engineering, Guangxi University for Nationalities, Nanning 530006, China*

[3]*Guangxi Key Laboratory of Hybrid Computation and IC Design Analysis, Nanning 530006, China*

[4]*Science Computing and Intelligent Information Processing of GuangXi higher education key laboratory，Nanning 530023, China*

Received 1 March 2014, www.tsi.lv

Abstract

An event structure acts as a denotational semantic model of concurrent systems. Action refinement is an essential operation in the design of concurrent systems. But there exists an important problem about preserving equivalence under action refinement. If two processes are equivalent with each other, we hope that they still can preserve equivalence after action refinement. In linear time equivalence and branching time equivalence spectrum, interleaving equivalences, which include interleaving trace equivalence and interleaving bisimulation equivalence are not preserved under action refinement [9-11, 14, 16, 21]. In this paper, we define a class of concurrent processes with specific properties and put forward the concept of clustered action transition, which ensures that interleaving equivalences are able to preserve under action refinement.

Keywords: event structure, action refinement, concurrency, interleaving equivalence, clustered equivalence

1 Introduction

In order to model concurrent systems, we hope to have formal method for hierarchical structure. Action refinement is the core operation of the hierarchical method, which interprets an action in higher abstract layer with a process in lower layer, hence reduces the level of abstraction and eventually reaches its implementation layer. In the development course from top to bottom of concurrent system, we must first build models, which depict the system with description language of top layer; subsequently, according to these descriptions, we complete its implementation. This course often requires equivalence notion to verify the correctness of implementation of system. More concretely, assuming that $\mathcal{P}$ represents the descriptions of system and $\mathcal{Q}$ represents its implementation, if $\mathcal{P}$ is equivalent with $\mathcal{Q}$ (expressed as $\mathcal{P} \approx \mathcal{Q}$), then this shows that $\mathcal{Q}$ is correct. In development, the description $\mathcal{P}$ of a system can be refined layer-by-layer, accordingly its implementation $\mathcal{Q}$ can be converted from framework into code or electronic components. Only the description and its implementation at all levels are required to maintain equivalence so as to ensure correctness of its implementation. This leads to an important question what kind of equivalence is maintained under action refinement, that is, if two concurrent systems are equivalent with each other, we hope that they still can preserve equivalence after action refinement.

Vogler [14, 15] first raised the basic thought of preserving equivalence under action refinement. Czaja, Van Glabbeek and Goltz [21] demonstrated that if interleaving bisimulation equivalence doesn't produce choice operations or action self-concurrences after actions are refined then it can preserve equivalence under action refinement, but interleaving trace equivalence still cannot preserve. Goltz and Wehrheim [20] proved that history preserving bisimulation is consistent with global causal dependencies, but they did not further discuss about the problem how to preserve equivalence under action refinement, and did not discuss that there are other situations under environment of action independencies. Van Glabbeek and Goltz [11, 21] summarized the research results of action refinement, gave a detailed explanation for preserving equivalence problem under action refinement, and proved that interleaving equivalence cannot preserve under action refinement in general, but did not discussed further. Moreover, no work further discusses preserving problem under action refinement of interleaving trace equivalence and interleaving bisimulation equivalence. In this paper, we define a class of concurrent processes with specific properties and put forward the concept of clustered action transition, which ensures that interleaving equivalences are able to preserve under action refinement in the absence of constraints.

[*] ***Corresponding author*** e-mail: himrwujz@126.com

2 Event structures and action refinement

Let Act be a set of actions.

Definition 2.1 [2, 12, 18]: A event structure $\mathcal{P}$ is a 5-tuple $(E,<,\#,\Delta,l)$, where

- E is the set of events;
- $<\subseteq E\times E$ is irreflexive partial relation, and satisfy the rule of finite causes that $\forall e\in E:\{e1\in E|e1<e\}$ is finite; In addition, its inverse "<" is expressed as ">";
- $\#\subseteq E\times E$ is irreflexive and finite conflicting relation, and satisfy the rule of inheriting of conflict that $\forall e1,e2,e3\in E: e1<e2\wedge e1\#e3\Rightarrow e2\#e3$;
- $\Delta\subseteq E\times E$ is irreflexive concurrent relation, altogether with < and # to satisfy the principle of partition that $<\bigcup\#\bigcup\Delta=E\times E$, $e1\Delta e2\Leftrightarrow\neg(e1=e2\vee e1<e2\vee e2<e1\vee e1\#e2)$;
- $l:E\to Act$ is a label function of actions.

In this paper, let $\mathbb{S}$ denote the set of all event structures.

Definition 2.2 [2, 11]: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called isomorphic (expressed as $\mathcal{P}\cong\mathcal{Q}$) if there exists an bijection between their sets and preserves corresponding relations with <, #, Δ and same corresponding labels.

Unless specified, we do not discriminate isomorphic event structures.

The behaviours of a event structure are expressed with its configurations which are considered as all possible states of the system. The following is its definition.

Definition 2.3 [2, 12, 18]:Let X be a subset of the set $E_{\mathcal{P}}$ of all events in event structure $\mathcal{P}$.

(1) X is left closed if $\forall e1,e\in E: e\in X\wedge e1<e\Rightarrow e1\in X$;

(2) X is conflicted-free if $\mathcal{P}|_X$ is conflicted-free;

(3) X is a configuration if X is not only left closed but also conflicted-free.

Here, let $C(\mathcal{P})$ represent the set of all configurations in event structure $\mathcal{P}$.

A configuration X ($X\in C(\mathcal{P})$) is called (successfully) terminated configuration if $\forall e\in E: e\notin X\Rightarrow\exists e1\in X: e1\#e$.

The event structure is also often represented with graph, where $\to, \cdots$ stands for casual relation and immediately conflict relation in event structure respectively, inherited conflict relation is not considered and indenpent relation is not explicitly expressed.

Example 2.1: A system $\mathcal{P}=(a|b)+(c;b;d)$, executing either a, b concurrently or c, b and d sequentially, can be described by the event structure with events $e1$, $e2$, $e3$, $e4$, $e5$ with $l(e1)=a$, $l(e3)=c$, $l(e2)=l(e4)=b$, $l(e5)=d$, where $e1\Delta e2$, $e3<e4<e5$, each of $e1$, $e2$ is in conflict with each of $e3$, $e4$, $e5$. This event structure is expressed in Figure 1.

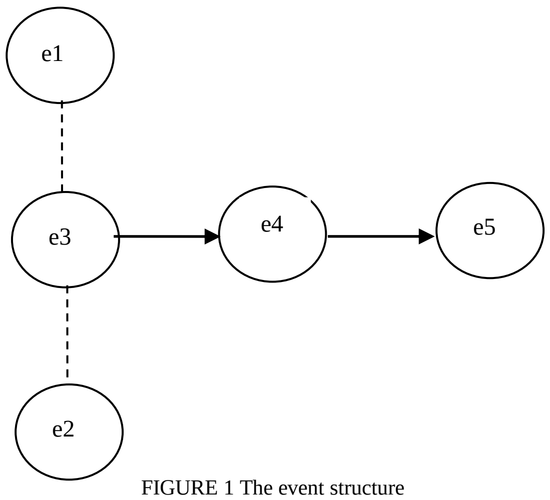

FIGURE 1 The event structure

Its configurations are $\varnothing,\{e1\},\{e2\}$, $\{e1,e2\},\{e3\}$, $\{e3,e4\}$, $\{e3,e4,e5\}$, where $\{e1,e2\}$, $\{e3,e4,e5\}$ are terminated configurations.

Main thought of action refinement [1, 3, 5, 8, 22] is: replace an action in higher layer with a process in lower layer, do in the same way layer by layer, until get detailed design or implementation of the system.

Definition 2.4 [13, 14, 17, 22]: A function $ref: Act\to E-\{\varnothing\}$ is called a refinement function of event structure, if $\forall a\in Act: ref(a)$ is not empty, finite and conflict-free. Let $\mathcal{P}\in\mathbb{S}$. $ref(\mathcal{P})$ is an event structure defined as follows:

$$E_{ref(\mathcal{P})}=\left\{(e,e')\middle|e\in E_{\mathcal{P}},e'\in E_{ref(l_{\mathcal{P}}(e))}\right\},$$

$(e1,e1')<_{ref(\mathcal{P})}(e2,e2')$ *iff* $e1<_{\mathcal{P}}e2$ or

$e1=e2\wedge e1'<_{ref(l_{\mathcal{P}}(e1))}e2'$.

$(e1,e1')\#_{ref(\mathcal{P})}(e2,e2')$ *iff* $e1\#_{\mathcal{P}}e2$,

$(e1,e1')\Delta_{ref(\mathcal{P})}(e2,e2')$ *iff* $e1\Delta_{\mathcal{P}}e2$ or

$e1=e2\wedge e1'\Delta_{ref(l_{\mathcal{P}}(e1))}e2'$,

$l_{ref(\mathcal{P})}(e,e')=l_{ref(l_{\mathcal{P}}(e))}(e')$.

Example 2.2: Continue with Example 2.1. Assuming that $ref(b)=(b1;b2)+b3$, the event structure after action refinement is expressed in Figure 2.

Under action refinement, each event e labelled by the action b is replaced by a disjoint copy, $\mathcal{P}_e$, of $ref(b)$, i.e., the event $e2$ is replaced by ($e22$<$e23$)#$e21$ and the event $e4$ is replaced by ($e41$<$e42$)#$e43$ [17]. The causality and conflict structure is inherited from $\mathcal{P}$: all events which were casually before e will be casually before all events of $\mathcal{P}_e$, every event which casually followed e will casually follow all events of $\mathcal{P}_e$, and all events in conflict with e will be in conflict with all events of $\mathcal{P}_e$.

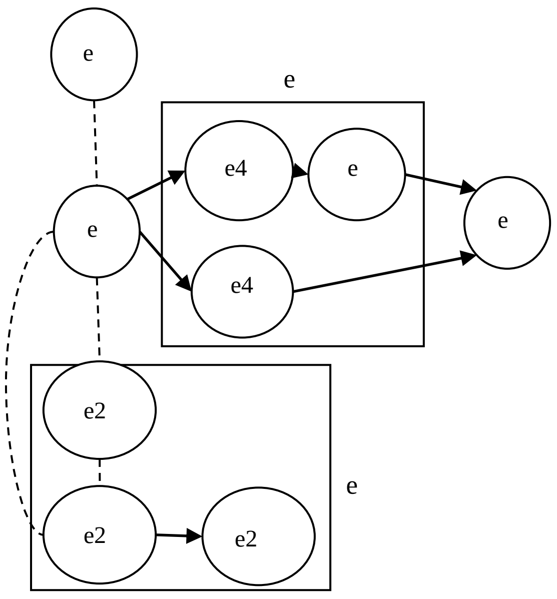

FIGURE 2 The event structure after action refinement

3 Interleaving equivalences

Interleaving equivalences can be divided into interleaving trace equivalence and interleaving bisimulation equivalence [9, 10, 11, 21]. To begin with, we give the definition of single action transition.

Definition 3.1: Let $\mathcal{P} \in \mathbb{S}$. A transition relation $X \xrightarrow{a}_{\mathcal{P}} X'$ is called single action transition if $a \in Act, X, X' \in C(\mathcal{P}), X \subseteq X'$, and $\exists e \in E_{\mathcal{P}}$: $X' - X = e, l_{\mathcal{P}}(e) = a$.

Here, $X \xrightarrow{a}_{\mathcal{P}} X'$ denotes that the state expressed by configuration X turns into the one expressed by configuration X' after performing action a in event structure $\mathcal{P} \in \mathbb{S}$.

Then, we define trace and interleaving trace equivalence.

Definition 3.2: Let $\mathcal{P} \in \mathbb{S}$. A word $w = a_1 \cdots a_n \in Act^*$ is called a trace of event structure $\mathcal{P}$ if $\exists X_0, \cdots, X_n \in C(\mathcal{P}) : X_0 = \varnothing$ and $X_{i-1} \xrightarrow{a_i} X_i$, $i = 1, \cdots, n$.

Here, $trs(\mathcal{P})$ represents the set of all traces in event structure $\mathcal{P}$.

Definition 3.3: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$. A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called interleaving trace equivalence (expressed as $\mathcal{P} \approx_{it} \mathcal{Q}$) if $trs(\mathcal{P})=trs(\mathcal{Q})$.

Furthermore, we define interleaving bisimulation equivalence.

Definition 3.4: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$. A relation $R \subseteq C(\mathcal{P}) \times C(\mathcal{Q})$ is called a interleaving bisimulation between $\mathcal{P}$ and $\mathcal{Q}$ if $(\varnothing, \varnothing) \in R$ and if $(X, Y) \in R$ then

$X \xrightarrow{a}_{\mathcal{P}} X', a \in Act \Rightarrow \exists Y' : Y \xrightarrow{a}_{\mathcal{Q}} Y' \wedge (X', Y') \in R$,

$Y \xrightarrow{a}_{\mathcal{Q}} Y', a \in Act \Rightarrow \exists X' : X \xrightarrow{a}_{\mathcal{P}} X' \wedge (X', Y') \in R$.

A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called interleaving bisimulation equivalence (expressed as $\mathcal{P} \approx_{ib} \mathcal{Q}$) if there exists a interleaving bisimulation between them.

Finally, we introduce another kind of equivalence named pomset trace equivalence which depends on partial order theory. we will still study whether quotient event structure and original event structure are pomset trace equivalences or not.

Definition 3.5: Let $\mathcal{P} \in \mathbb{S}$.

(1) Let $R1 = \langle X1, <_{X1}, l|_{X1} \rangle$ and $R2 = \langle X2, <_{X2}, l|_{X2} \rangle$ be two partial sets labelled in Act $R1$ and $R2$ is isomorphic with each other (expressed as $R1 \cong_p R2$) if there exists a bisection $f : X \to Y$ such that $\forall e1, e2 \in X1 : e1 <_{X1} e2 \Leftrightarrow f(e1) <_{X2} f(e2)$, and $l|_{X1} = l|_{X2} \circ f$. A isomorphic class of partial set in the set Act is called pomset.

(2) Partial sets of configuration X is $pomset(X) = \left[\left(X, <_X, l|_X \right) \right]_{\cong_p}$. The set of all pomsets in event structure $\mathcal{P}$ is $pomsets(\mathcal{P}) = \left\{ pomset(X) \middle| X \in C(\mathcal{P}) \right\}$.

Definition 3.6: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$. Assume that $pomsets(\mathcal{P}) = \left\{ pomset(X) \middle| X \in C(\mathcal{P}) \right\}$ and $pomsets(\mathcal{Q}) = \left\{ pomset(Y) \middle| Y \in C(\mathcal{Q}) \right\}$.

A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called pomset trace equivalence (expressed as $\mathcal{P} \approx_{pt} \mathcal{Q}$) if $pomsets(\mathcal{P}) = pomsets(\mathcal{Q})$.

Article [17] has proved that pomset trace equivalence is able to preserve under action refinement.

Proposition 3.1: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$, let ref be a refinement function. If $\mathcal{P} \approx_{pt} \mathcal{Q}$ then $\text{ref}(\mathcal{P}) \approx_{pt} \text{ref}(\mathcal{Q})$.

The proof confers theorem 8.1 in article [11].

Pomset trace equivalence requires that casual and independent relations between events must be consistent. Apparently, it can directly reach a conclusion that pomset trace equivalence implies interleaving trace equivalence.

Proposition 3.2: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. If $\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$ then $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}$.

Subsequently, we introduce distinct kind of equivalence called history preserving bisimulation which is finer than interleaving bisimulation and is able to preserve under action refinement.

Definition 3.7: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. A relation $R\subseteq C(\mathcal{P})\times C(\mathcal{Q})\times\mathcal{P}(E_{\mathcal{P}}\times E_{\mathcal{Q}})$ is called a history preserving bisimulation between $\mathcal{P}$ and $\mathcal{Q}$ if $(\varnothing,\varnothing,\varnothing)\in R$, there always exists $(X,Y,f)\in R$, then f is a bijection between X and Y;

$X\xrightarrow{a}_{\mathcal{P}}X',a\in Act\Rightarrow\exists Y',f'$:

$Y\xrightarrow{a}_{\mathcal{Q}}Y',(X',Y',f')\in R\wedge f'|_X=f$;

$Y\xrightarrow{a}_{\mathcal{Q}}Y',a\in Act\Rightarrow\exists X',f'$:

$X\xrightarrow{a}_{\mathcal{P}}X',(X',Y',f')\in R\wedge f'|_X=f$.

Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called history preserving equivalence (expressed as $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$) if there exists a history preserving bisimulation between them.

History preserving equivalence is able to preserve under action refinement. The following gives the conclusion by means of Proposition 3.3.

Proposition 3.3: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$, let ref be a refinement function. If $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$ then $\mathrm{ref}(\mathcal{P})\approx_{\mathrm{h}}\mathrm{ref}(\mathcal{Q})$.

The proof can confer theorem 9.1 and proposition 9.2 in article [11].

According to definition 3.4 and definition 3.7, we immediately attain a result that history preserving equivalence can deduce interleaving bisimulation equivalence.

Proposition 3.4: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. If $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$ then $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}$.

Interleaving equivalences cannot preserve under action refinement. However, if restrictions with some properties based on event structure model are placed, some useful results will be attained. The following proposition shows that if no independency exists in event structure then interleaving equivalences (include interleaving trace equivalence and interleaving bisimulation equivalence) are able to preserve under action refinement.

Proposition 3.5: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$, let ref be a refinement function $\Delta_{\mathcal{P}}=\Delta_{\mathcal{Q}}=\varnothing$.

(1) If $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}$ then $\mathrm{ref}(\mathcal{P})\approx_{\mathrm{it}}\mathrm{ref}(\mathcal{Q})$.

(2) If $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}$ then $\mathrm{ref}(\mathcal{P})\approx_{\mathrm{ib}}\mathrm{ref}(\mathcal{Q})$.

Proof:

(1) For any configuration $X\in C(\mathcal{P})$, by definition 2.2, there is $\#_{\mathcal{P}|_X}=\varnothing$. Moreover, because $\Delta_{\mathcal{P}}=\varnothing$, X is left closed, there exists a total order relation in *X*. By definition 3.2, X corresponds to unique trace (expressed as w_X), namely, $\forall X,Y\in C(\mathcal{P})\wedge X\neq Y\Rightarrow\exists w_X,w_Y\in trs(\mathcal{P})\wedge w_X\neq w_Y$.

On the condition that actions names cannot give rise to confusion, we represent corresponding event with its action name, $pomsets(\mathcal{P})=trs(\mathcal{P})$. Similarly, $pomsets(\mathcal{Q})=trs(\mathcal{Q})$. Given $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}$, namely, $trs(\mathcal{P})=trs(\mathcal{Q})$, we obtain $pomsets(\mathcal{P})=pomsets(\mathcal{Q})$, hence $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$. By proposition 3.1, $\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}\Rightarrow\mathrm{ref}(\mathcal{P})\approx_{\mathrm{pt}}\mathrm{ref}(\mathcal{Q})$. Also, by proposition 3.2, we obtain that $\mathrm{ref}(\mathcal{P})\approx_{\mathrm{pt}}\mathrm{ref}(\mathcal{Q})\Rightarrow\mathrm{ref}(\mathcal{P})\approx_{\mathrm{it}}\mathrm{ref}(\mathcal{Q})$.

(2) Based on the above proof, For any configuration $X\in C(\mathcal{P})$, there only exists a total order relation in X. Similarly, for any configuration $Y\in C(\mathcal{Q})$, So is *Y*. Due to $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}$, we obtain $X\xrightarrow{a}_{\mathcal{P}}X',a\in Act\Rightarrow\exists Y':Y\xrightarrow{a}_{\mathcal{Q}}Y'\wedge(X',Y')\in R$ and $(\varnothing,\varnothing)$, (X,Y), $(X',Y')\in R$. Now, we want to prove $X\cong_p Y$ and $X'\cong_p Y'$. Distinctly, the bisimulation between $\mathcal{P}$ and $\mathcal{Q}$ starts from initial configuration $\varnothing$ and gets to states of bisimulation (expressed as *X*, *Y*) by performing same actions. Also, *X*, *Y* is total order and they execute corresponding events of same actions, hence $X\cong_p Y$. Similarly, $X'\cong_p Y'$. According to symmetry [6, 19], for any $Y\xrightarrow{a}_{\mathcal{Q}}Y',a\in Act\Rightarrow\exists X':X\xrightarrow{a}_{\mathcal{P}}X'\wedge(X',Y')\in R$, there also exist $Y\cong_p X$ and $Y'\cong_p X'$. Following the above method, we are able to find a history preserving bisimulation between $\mathcal{P}$ and $\mathcal{Q}$, i.e., $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$. Also, by proposition 3.3, $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}\Rightarrow\mathrm{ref}(\mathcal{P})\approx_h\mathrm{ref}(\mathcal{Q})$. By proposition 3.4, $\mathrm{ref}(\mathcal{P})\approx_h\mathrm{ref}(\mathcal{Q})\Rightarrow\mathrm{ref}(\mathcal{P})\approx_{ib}\mathrm{ref}(\mathcal{Q})$.

4 Clustered action transition

We present a new class of action transition, where A is a multiple set in action set Act and all actions within A independently perform with each other. We call this multiple set A as a clustered action and call this class transitions as clustered action transitions. With clustered action transitions, we construct two new types of equivalence.

Definition 4.1: Let $\mathcal{P}\in\mathbb{S}$. A transition $X\xrightarrow{A}X'$ is called a clustered action transition if $A\in N^{Act}$ (i.e., A is multiple set in Act)

$X, X' \in C(\mathcal{P}), X \subseteq X', X' - X = G$, the events in set G satisfy:

(1) entire independency of causes:

$$\forall d, e \in G : (d \ \Delta_{X'} \ e) \wedge \left(\{e_1 \in E_{\mathcal{P}} | e_1 \ \Delta_{\mathcal{P}} \ e\} \cup \{e\} = \{e_2 \in E_{\mathcal{P}} | e_2 \ \Delta_{\mathcal{P}} \ d\} \cup \{d\} = G\right)$$ and $l_{\mathcal{P}}(G) = A$.

(2) same causality:

$$\forall e_1 \in E_{\mathcal{P}} \setminus G, \exists e_2 \in G : (e_1 < e_2 \Rightarrow \forall e_3 \in G : e_1 < e_3) \vee (e_1 > e_2 \Rightarrow \forall e_3 \in G : e_1 > e_3);$$

(3) same conflict relation:

$$\forall e_1 \in E_{\mathcal{P}} \setminus G, \exists e_2 \in G : (e_1 \# e_2 \Rightarrow \forall e_3 \in G : e_1 \# e_3).$$

Here, $l_{\mathcal{P}}(G) \in N^{Act}$ results from

$$l_{\mathcal{P}}(G)(a) = \left|\{e \in G | l_{\mathcal{P}}(e) = a\}\right|.$$

Clustered action transition $X \xrightarrow{A} X'$ represents that after independently executing all actions of set A, the state expressed by configuration X is changed into the one expressed by configuration X' in event structure $\mathcal{P}$.

Definition 4.2: suppose that e is an event in event structure $\mathcal{P}$. Independency set of cause on e $\varphi(e)$, including e itself, is defined as $\varphi(e) = \{e_1 \in E_{\mathcal{P}} | e_1 \ \Delta_{\mathcal{P}} \ e\} \cup \{e\}$.

In order to study the follow-up problems, we give a proposition in advance.

Proposition 4.1: Let $\mathcal{P} \in \mathbb{S}$. If all transitions in $\mathcal{P}$ are clustered action transitions then

(1) $\forall d \in E_{\mathcal{P}} \Rightarrow \exists (X \xrightarrow{A} X') : \varphi(d) = G$, where $A \in N^{Act}$, $X, X' \in C(\mathcal{P}), X \subseteq X', X' - X = G$ and $l_{\mathcal{P}}(G) = A$.

(2) $\forall (X \xrightarrow{A} X') \Rightarrow \exists d \in E_{\mathcal{P}} : G = \varphi(d)$, where $A \in N^{Act}$, $X, X' \in C(\mathcal{P}), X \subseteq X', X' - X = G$ and $l_{\mathcal{P}}(G) = A$;

(3) $\bigcup_{e \in E_{\mathcal{P}}} \varphi(e) = E_{\mathcal{P}}$;

(4) $\forall e_1, e_2 \in E_{\mathcal{P}} : e_1 \neq e_2 \wedge \neg(e_1 \ \Delta \ e_2) \Rightarrow \varphi(e_1) \cap \varphi(e_2) = \varnothing$.

Proof: Conclusions (1), (2) and (3) is very easy to reach, omitted here. Only give proof of (4). If $\varphi(e_1) \cap \varphi(e_2) \neq \varnothing$ then $\exists e \in \varphi(e_1) \cap \varphi(e_2)$. By definition 4.1, we obtain $\varphi(e) = \varphi(e_1) = \varphi(e_2)$, consequently obtain *either* $e_1 \Delta e_2$ or $e_1 = e_2$. This contradict with $e_1 \neq e_2 \wedge \neg(e_1 \ \Delta \ e_2)$. Accordingly, arrive at the conclusion that

$\forall e_1, e_2 \in E_{\mathcal{P}} : e_1 \neq e_2 \wedge \neg(e_1 \ \Delta \ e_2) \Rightarrow \varphi(e_1) \cap \varphi(e_2) = \varnothing$.

The above proposition shows that if all transitions in an event structure are clustered action transitions, then all independent actions involved in a clustered action transition are seen as a “big” action, its corresponding events can be thought of as a “big” event. Hence, not only no independency between events exists in this event structure but also independency sets of cause divide set of events into different parts, which induces an equivalence relation in set of events of this event structure.

Definition 4.3: Let $\mathcal{P} \in \mathbb{S}$. A sequence $w = A_1 \cdots A_n \ \left(A_i \in N^{Act} \ (i = 1, \cdots, n)\right)$ is called a clustered trace of event structure $\mathcal{P}$ if $\exists X_0, \cdots, X_n \in C(\mathcal{P}) : X_0 = \varnothing$ and $X_{i-1} \xrightarrow{A_i} X_i, i = 1, \cdots, n$ is a clustered action transition.

Here, $Clusteredtrs(\mathcal{P})$ represents the set of all clustered traces of event structure $\mathcal{P}$.

Subsequently, we define clustered trace equivalence.

Definition 4.4: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$, all transitions in $\mathcal{P}$ and $\mathcal{Q}$ be clustered transitions. Let $Clusteredtrs(\mathcal{P})$ and $Clusteredtrs(\mathcal{Q})$ be the sets of all clustered traces of $\mathcal{P}$, $\mathcal{Q}$ respectively. A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called clustered trace equivalence (expressed as $\mathcal{P} \approx_{ct} \mathcal{Q}$) if $Clusteredtrs(\mathcal{P}) = Clusteredtrs(\mathcal{Q})$.

The following proposition shows that clustered trace equivalence can imply interleaving trace equivalence.

Proposition 4.2: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$. If all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ are clustered action transitions then $\mathcal{P} \approx_{ct} \mathcal{Q} \Rightarrow \mathcal{P} \approx_{it} \mathcal{Q}$.

Proof: By definition 4.1 and proposition 4.1, $\forall A_i, A_j (i = 1, ..., n, \ j = 1, ..., n, \ i \neq j)$ in a clustered trace, then all actions in A_i are independent of those in A_j. Performing of any action in A_i does not interfere with those in A_j, vice versa. Consequently, $Clusteredtrs(\mathcal{P}) = Clusteredtrs(\mathcal{Q}) \Rightarrow trs(\mathcal{P}) = trs(\mathcal{Q})$, i.e. $\mathcal{P} \approx_{ct} \mathcal{Q} \Rightarrow \mathcal{P} \approx_{it} \mathcal{Q}$.

Provide that $W = A_1, ..., A_n$, where $A_i \in N^{Act}$ $(i = 1, ..., n)$ be a clustered trace, and $|A_i|$ stand for the number of elements, then the clustered trace W corresponds to $|A_1|! \times |A_2|! \times ... \times |A_n|!$ general traces.

After discussing clustered trace equivalence, we begin with another new class of equivalence and study whether they can maintain under action refinement or not. This class of equivalence is designated clustered bisimulation equivalence. The following provides for its definition.

Definition 4.5: Let $\mathcal{P}, \mathcal{Q} \in \mathbb{S}$, let all transitions in $\mathcal{P}$ and $\mathcal{Q}$ be clustered action transitions. A relation $R \subseteq C(\mathcal{P}) \times C(\mathcal{Q})$ is called a clustered bisimulation between $\mathcal{P}$ and $\mathcal{Q}$ if $(\varnothing,\varnothing)\in R$ and if $(X,Y)\in R$ then

$$X \xrightarrow{A}_{\mathcal{P}} X', A\in N^{Act} \Rightarrow \exists Y': Y \xrightarrow{A}_{\mathcal{Q}} Y' \wedge (X',Y')\in R,$$

$$Y \xrightarrow{A}_{\mathcal{Q}} Y', A\in N^{Act} \Rightarrow \exists X': X \xrightarrow{A}_{\mathcal{P}} X' \wedge (X',Y')\in R.$$

A relation between $\mathcal{P}$ and $\mathcal{Q}$ is called clustered bisimulation equivalence (expressed as $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}$) if there exists a clustered bisimulation between them.

Obviously, by definition 4.5, we come to a decision that $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}$.

Proposition 4.3 Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. If all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ are clustered action transitions then $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}$.

Procedure of proof is similar to that of proposition 4.2, omitted here.

5 Preserving of interleaving equivalences

These papers [9-11, 14, 16, 21] have demonstrated that interleaving equivalence cannot preserve under action refinement in the general case. However, proposition 3.5 shows that interleaving equivalence without concurrency can preserve under action refinement. In this part, we will testify that in the presence of concurrency interleaving equivalence is able to preserve under action refinement in given conditions. In order to prove the conclusion, we first discuss the relationship between clustered trace equivalence and partial order trace equivalence.

Proposition 5.1: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. If all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ are clustered action transitions, then $\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$.

Proof: For any configuration $X\in C(\mathcal{P})$, since all transitions in event structures $\mathcal{P}$ are clustered action transitions, there exists unique one clustered trace, let it be *W*, construction of *W* starts from initial configuration $\varnothing$ and evolves into configuration *X*. Also, given $\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}$, we obtain $\mathit{Clusteredtrs}(\mathcal{P})=\mathit{Clusteredtrs}(\mathcal{Q})$, then there exists a configuration $Y\in C(\mathcal{Q})$ reached by the implementation of clustered trace *W*. Obviously, $X\cong_p Y$, i.e. $\forall X\in C(\mathcal{P})\Rightarrow \exists Y\in C(\mathcal{Q})\wedge X\cong_p Y$. Similarly, we obtain $\forall Y\in C(\mathcal{Q})\Rightarrow\exists X\in C(\mathcal{P})\wedge Y\cong_p X$. Isomorphism ensures they have same labels, therefore $\mathit{pomsets}(\mathcal{P})=\mathit{pomsets}(\mathcal{Q})$ i.e. $\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$.

At present, we discuss about the problem that interleaving equivalence can preserve under action refinement in given conditions.

Proposition 5.2: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. Assume that all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ be clustered action transitions, *ref* be a refinement function. If $\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}$ then $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}\Rightarrow \mathit{ref}(\mathcal{P})\approx_{\mathrm{it}}\mathit{ref}(\mathcal{Q})$.

Proof: By proposition 5.1, $\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$. Also, by proposition 3.1, $\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}\Rightarrow \mathit{ref}(\mathcal{P})\approx_{\mathrm{pt}}\mathit{ref}(\mathcal{Q})$. And, by proposition 3.2, $\mathit{ref}(\mathcal{P})\approx_{\mathrm{pt}}\mathit{ref}(\mathcal{Q})\Rightarrow \mathit{ref}(\mathcal{P})\approx_{\mathrm{it}}\mathit{ref}(\mathcal{Q})$. Consequently, we arrive at a conclusion that $\mathcal{P}\approx_{\mathrm{it}}\mathcal{Q}\Rightarrow \mathit{ref}(\mathcal{P})\approx_{\mathrm{it}}\mathit{ref}(\mathcal{Q})$.

This proposition shows that if two event structures satisfy clustered trace equivalence and their all transitions are clustered action transitions, then interleaving equivalence can hold under action refinement. This is an important conclusion and is one of research goals to be achieved.

Subsequently, we discuss the topic that interleaving bisimulation equivalence can preserve under action refinement in given conditions.

Proposition 5.3: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. If all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ are clustered action transitions then $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$.

Proof: Assume that the relation $R\subseteq C(\mathcal{P})\times C(\mathcal{Q})$ be a clustered bisimulation between $\mathcal{P}$ and $\mathcal{Q}$. We construct a history preserving bisimulation $\tilde{R}\subseteq C(\mathcal{P})\times C(\mathcal{Q})$ between $\mathcal{P}$ and $\mathcal{Q}$ by inductive method:

(1) First let the element $(\varnothing,\varnothing)$ add into the set $\tilde{R}$;

(2) Now assume that (X,Y) have already been an element of $\tilde{R}$, and given $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}$, also assume that $X\xrightarrow{A}_{\mathcal{P}}X', Y\xrightarrow{A}_{\mathcal{Q}}Y', A\in N^{Act}$ and $(X,Y)\in R$, also because of $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}\Rightarrow\mathcal{P}\approx_{\mathrm{ct}}\mathcal{Q}$, by definition 4.3, we obtain $\mathcal{P}\approx_{\mathrm{pt}}\mathcal{Q}$. Hence, $X\cong_p Y$ and $X'\cong_p Y'$. If there are n (n>1) elements in multiple action set *A* and the execution among these elements is interleaving then there exist n! sequences of single action transition: $X\xrightarrow{a_1}_{\mathcal{P}}X_1\xrightarrow{a_2}_{\mathcal{P}}X_2\cdots X_{n-1}\xrightarrow{a_{n-1}}_{\mathcal{P}}X'$.

Accordingly, there exists the corresponding sequence of single action transition: $Y\xrightarrow{a_1}_{\mathcal{Q}}Y_1\xrightarrow{a_2}_{\mathcal{Q}}Y_2,\dots,Y_{n-1}\xrightarrow{a_{n-1}}_{\mathcal{Q}}Y'$, and meets $X_1\cong_p Y_1, X_2\cong_p Y_2,\dots,X_{n-1}\cong_p Y_{n-1}$. Apparently, add these n elements $(X_i,Y_i)(i=1,\dots,n)$ to $\tilde{R}$, meanwhile, add (X',Y') to $\tilde{R}$, where $X_i\cong_p Y_i$. Along this route, we construct all elements of the set $\tilde{R}$ which are configuration pairs acquired by same single action transitions from *X* and *Y* respectively. Therefore, $\tilde{R}$ is a history preserving bisimulation, i.e. $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$.

This proposition shows that clustered bisimulation equivalence $\approx_{\mathrm{bb}}$ can imply history preserving equivalence $\approx_{\mathrm{h}}$.

Subsequently, we discuss the relationship under action refinement between clustered bisimulation equivalence and interleaving bisimulation equivalence.

Proposition 5.4: Let $\mathcal{P},\mathcal{Q}\in\mathbb{S}$. Assume that all transitions in event structures $\mathcal{P}$ and $\mathcal{Q}$ be clustered action transitions, *ref* be a refinement function. If $\mathcal{P}\approx_{\mathrm{cb}}\mathcal{Q}$ then $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}\Rightarrow ref(\mathcal{P})\approx_{\mathrm{ib}} ref(\mathcal{Q})$.

Proof: By proposition 5.3, $\mathcal{P}\approx_{\mathrm{h}}\mathcal{Q}$. Also, by proposition 3.3, $ref(\mathcal{P})\approx_{\mathrm{h}} ref(\mathcal{Q})$. And, by proposition 3.4, $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}\Rightarrow ref(\mathcal{P})\approx_{\mathrm{ib}} ref(\mathcal{Q})$. Therefore, we can obtain $\mathcal{P}\approx_{\mathrm{ib}}\mathcal{Q}\Rightarrow ref(\mathcal{P})\approx_{\mathrm{ib}} ref(\mathcal{Q})$.

This proposition shows that if two event structures satisfy clustered bisimulation equivalence and their all transitions are clustered action transitions, then interleaving bisimulation equivalence also can hold under action refinement. Proposition 5.2 and proposition 5.4 extend proposition 3.5, which demonstrates that in the presence of concurrency interleaving equivalence is able to preserve under action refinement in given conditions.

6 Results and Discussion

The paper has demonstrated that

(1) In event structures without independency between events, interleaving trace equivalence and interleaving bisimulation equivalence can preserve under action refinement.

(2) In event structures, if all transitions are clustered action transitions, then with clustered trace equivalence between event structures, we can reach that interleaving trace equivalence can preserve under action refinement; likewise, with clustered bisimulation equivalence between event structures, we can reach that interleaving bisimulation equivalence can preserve under action refinement. Therefore, we find a class of concurrent processes with specific properties, which enable interleaving equivalence to preserve under action refinement in the absence of constraint.

Our next work is to introduce the thought of clustered action transition into model checking so as to deal with states explosion problem [4, 7, 15] in the process of system verification.

Acknowledgements

This Work is partly supported by Grants No. HCIC201306 of Guangxi HCIC lab Open Fund, the National Natural Science Foundation of China under Grant No. 11371003, the Natural Science Foundation of Guangxi under Grant No. 2011GXNSFA018154 and No. 2012GXNSFGA060003, and Science Computing and Intelligent Information Processing of GuangXi higher education key laboratory No. GXSCIIP201201.

References

[1] Wu J 2001 Action refinement in timed LOTOS *Proc Of ASCM'01 World Scientific Publ* 183-92
[2] Wang Y, Wu J, Jiang J 2007 Process algebra-symmetry and action refinement,*Chinese Science Press*06
[3] Majster-Cederbaum M, Wu J 2003 Towards action refinement for true concurrent real time *Acta Informatica* **39** 1-47
[4] Clarke E M, Grumberg O, Minea M, Peled D 1999 State space reduction using partial order techniques *STTT* **2**(3) 279-87
[5] Darondeau P, Degano P 1993 Refinement of actions in event structures and casual trees *Theoretical Computer Science* **118** 21-48
[6] Jiang J, Wu J 2005 Symmetry and autobisimulation *Proceedings of the 6th International Conference on Parallel and Distributed Computing, Applications and technologies IEEE Computer Society Press* 866-70
[7] Jiang J, Wu J, Yan W 2005 Structural reductions in process algebra languages *Proceedings of the 11th Joint International Computer Conference World Scientific Publishing Co* 596-600
[8] Degano P, Gorrierri R 1995 A causal operational semantics of action refinement *Information and Computation* **122** 97-119
[9] van Glabbeek R J, Goltz U 1989 Equivalence notions for concurrent systems and refinement of actions *Mathematical Foundations of Computer Science Lecture Notes in Computer Science* **379** 237-48
[10] van Glabbeek R J, Goltz U 1990 A deadlock-sensitive congruence for action refinement *Institut fuer Informatik Technische Universitaet Munchen SFB-Bericht* 342/23/90 A
[11] van Glabbeek R J, Goltz U 2001 Refinement of actions and equivalence notions for concurrent systems *Acta Informatica* **37**(4/5) 229-327
[12] Tang W, Wu J, Zheng D 2014 On fuzzy rough sets and their topological structures *Mathematical Problems in Engineering* 01
[13] Fecher H, Majster-Cederbaum M, Wu J 2002 Refinement of actions in a real-time process algebra with a true concurrency model *Electronic Notes in Theoretical Computer Science* **70**(3) 620-40
[14] Vogler W 1991 Bisimulation and action refinement *STACS'91 Lecture Note in Computer Science* **480** 309-21
[15] Vogler W 1992 Modular construction and partial order semantics of Petri nets *Lecture Note in Computer Science* **625**
[16] Huhn M 1996 Action refinement and property inheritance in systems of sequential agents *Concur'96 Lecture Notes in Computer Science* **1119** 639-54
[17] Majster-Cederbaum M, Wu J, Yue H 2006 Refinement of actions for real-time concurrent systems with causal ambiguity *Acta Informatica* **42**(6/7) 389-418
[18] Winskel G 1989 An Introduction to Event structures *Berlin Springer LNCS* **354** 364-97
[19] Wu J, Fecher H 2004 Symmetric structure in logic programming *Journal of Computer Science and Technology* **19**(6) 803-11
[20] Goltz U, Wehrheim H 1996 Modelling causality by dependency of actions in branching time semantics *Information Processing Letters* **59**(4) 179-84
[21] Czaja I, van Glabbeek R J, Goltz U 1992 Interleaving semantics and action refinement with atomic choice *Advances in Petri Nets Lecture Notes in Computer Science* **609** 89-107
[22] Gorrieri R, Rensink A 2001 Action Refinement *Bergstra J A Ponse A and Smolka S A editors Handbook of Process Algebra New York Elsevier Science* 1047-47

Authors

	Weidong Tang, born in February, 1968, Nanning China **Current position, grades:** Teacher, Associate professor **University studies:** Computer Software and Theory **Scientific interest:** Symbolic computation, formal verification
	Jinzhao Wu, born in October, 1965, Nanning China **Current position, grades:** Professor, Ph.D. **University studies:** Computer Software and Theory **Scientific interest:** Symbolic computation, automated reasoning, formal methods
	Meiling Liu, born in January, 1979, Nanning China **Current position, grades:** Teacher, Lecturer **University studies:** Computer Software and Theory **Scientific interest:** Data Mining, formal verification